Terms used **laws of concurrency** Found **9,749** of **201,798**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Algebraic laws for nondeterminism and concurrency
Matthew Hennessy, Robin Milner
January 1985 **Journal of the ACM (JACM)**, Volume 32 Issue 1
Publisher: ACM Press
Full text available: pdf(1.89 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Since a nondeterministic and concurrent program may, in general, communicate repeatedly with its environment, its meaning cannot be presented naturally as an input/output function (as is often done in the denotational approach to semantics). In this paper, an alternative is put forth. First, a definition is given of what it is for two programs or program parts to be equivalent for all observers; then two program parts are said to be observation congruent if they are, in all ...

2 CIRCAL and the representation of communication, concurrency, and time
George J. Milne
April 1985 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 7 Issue 2
Publisher: ACM Press
Full text available: pdf(2.02 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The CIRCAL calculus is presented as a mathematical framework in which to describe and analyze concurrent systems, whether hardware or software. The dot operator is used to compose CIRCAL descriptions, and it is this operator which permits the natural modeling of asynchronous and simultaneous behavior, thus allowing the representation and analysis of system timing properties such as those found in circuits. The CIRCAL framework uses an ...

3 Laws of programming
C. A. R. Hoare, I. J. Hayes, He Jifeng, C. C. Morgan, A. W. Roscoe, J. W. Sanders, I. H. Sorensen, J. M. Spivey, B. A. Sufrin
August 1987 **Communications of the ACM**, Volume 30 Issue 8
Publisher: ACM Press
Full text available: pdf(1.50 MB) Additional Information: full citation, abstract, references, citings, index terms

A complete set of algebraic laws is given for Dijkstra's nondeterministic sequential programming language. Iteration and recursion are explained in terms of Scott's domain

theory as fixed points of continuous functionals. A calculus analogous to weakest preconditions is suggested as an aid to deriving programs from their specifications.

4 Strategic directions in concurrency research

Rance Cleaveland, Scott A. Smolka
December 1996 **ACM Computing Surveys (CSUR)**, Volume 28 Issue 4
Publisher: ACM Press
Full text available: pdf(323.67 KB) Additional Information: full citation, references, citings, index terms

5 Deriving refactorings for AspectJ

Leonardo Cole, Paulo Borba
March 2005 **Proceedings of the 4th international conference on Aspect-oriented software development AOSD '05**
Publisher: ACM Press
Full text available: pdf(242.09 KB) Additional Information: full citation, abstract, references, citings, index terms

In this paper we present aspect-oriented programming laws that are useful for deriving refactorings for AspectJ. The laws help developers to verify if the transformations they define preserve behaviour. We illustrate that by deriving several AspectJ refactorings. We also show that our laws are useful for restructuring two Java applications with the aim of using aspects to modularize common crosscutting concerns.

Keywords: AspectJ, aspect-oriented programming, refactoring, separation of concerns

6 Program schemas with concurrency: execution time and hangups

Bruce P. Lester
January 1975 **Proceedings of the 2nd ACM SIGACT-SIGPLAN symposium on Principles of programming languages POPL '75**
Publisher: ACM Press
Full text available: pdf(515.23 KB) Additional Information: full citation, abstract, references, citings

A class of program schemas with concurrency is defined as a natural extension of the standard notion of sequential flow-chart-like schemas. The question is considered as to whether such a program schema may reach a premature termination (or "hangup") for some interpretation. It is shown that in general this question is undecidable; however, it is shown to be decidable for the class of free program schemas. And an algorithm for testing this property is presented with an upper time bound that grow ...

7 Distributed bisimulations

Ilaria Castellani, Matthew Hennessy
October 1989 **Journal of the ACM (JACM)**, Volume 36 Issue 4
Publisher: ACM Press
Full text available: pdf(1.81 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A new equivalence between concurrent processes is proposed. It generalizes the well-known bisimulation equivalence to take into account the distributed nature of processes. The result is a noninterleaving semantic theory; concurrent processes are differentiated from processes that are non-deterministic but sequential. The new equivalence, together with its observational version, is investigated for a subset of the language CCS, and various algebraic characterizations are obtained.

8

Branching time and abstraction in bisimulation semantics
Rob J. van Glabbeek, W. Peter Weijland
May 1996 **Journal of the ACM (JACM)**, Volume 43 Issue 3
Publisher: ACM Press

Full text available: pdf(4.16 MB) Additional Information: full citation, abstract, references, citings, index terms

In comparative concurrency semantics, one usually distinguishes between linear time and branching time semantic equivalences. Milner's notion of observatin equivalence is often mentioned as the standard example of a branching time equivalence. In this paper we investigate whether observation equivalence really does respect the branching structure of processes, and find that in the presence of the unobservable action &tgr; of CCS this is no ...

Keywords: abstraction, action refinement, bisimulation, branching time, concurrency, process algebra semantic equivalence

9 A formal definition of priority in CSP
C. J. Fidge
September 1993 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 15 Issue 4
Publisher: ACM Press

Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, index terms, review

The process models of Ada and occam are formally based on the CSP process algebra. However, for fine-tuning real-time performance, they include "prioritized" constructs that have no counterparts in CSP. These constructs therefore lack any formal definition, a situation that leaves room for misunderstandings. We extend CSP with a formal definition of the notion of priority. The definition is then used to assess the transputer implementation of priority in occam and the definition ...

Keywords: Ada, Communicating Sequential Processes, Priority, occam, real-time programming

10 1983 Invited address solved problems, unsolved problems and non-problems in concurrency
Leslie Lamport
August 1984 **Proceedings of the third annual ACM symposium on Principles of distributed computing PODC '84**
Publisher: ACM Press
Full text available: pdf(1.09 MB) Additional Information: full citation, abstract, citings, index terms

This is an edited transcript of a talk given at last year's conference. To preserve the flavor of the talk and the questions, I have done very little editing—mostly eliminating superfluous words and phrases, correcting especially atrocious grammar, and making the obvious changes needed when replacing slides by figures. The tape recorder was not functioning for the first few minutes, so I had to recreate the beginning of the talk.

11 Solved problems, unsolved problems and non-problems in concurrency
Leslie Lamport
October 1985 **ACM SIGOPS Operating Systems Review**, Volume 19 Issue 4
Publisher: ACM Press
Full text available: pdf(1.28 MB) Additional Information: full citation, abstract, citings

This is an edited transcript of a talk given at last year's conference. To preserve the flavor of the talk and the questions, I have done very little editing---mostly eliminating superfluous words and phrases, correcting especially atrocious grammar, and making the obvious changes needed when replacing slides by figures. The tape recorder was not functioning for the first few minutes, so I had to recreate the beginning of the talk.

12 Mobile safe ambients

Francesca Levi, Davide Sangiorgi

January 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 1

Publisher: ACM Press

Full text available: pdf(496.44 KB) Additional Information: full citation, abstract, references, citings, index terms

Two forms of interferences are individuated in Cardelli and Gordon's *Mobile Ambients* (MA): *plain interferences*, which are similar to the interferences one finds in CCS and n-calculus; and *grave interferences*, which are more dangerous and may be regarded as programming errors. To control interferences, the MA movement primitives are modified; the resulting calculus is called *Mobile Safe Ambients* (SA).The modification also has computational significance. In the MA in ...

Keywords: Mobility, behavioral equivalences, interferences

13 Logic of global synchrony

Yifeng Chen, J. W. Sanders

March 2004 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 26 Issue 2

Publisher: ACM Press

Full text available: pdf(265.65 KB) Additional Information: full citation, abstract, references, index terms

An intermediate-level specification formalism (i.e., specification language supported by laws and a semantic model), Logs, is presented for PRAM and BSP styles of parallel programming. It extends pre-post sequential semantics to reveal states at points of global synchronization. The result is an integration of the pre-post and reactive-process styles of specification. The language consists of only six commands from which other useful commands can be derived. Parallel composition is simply logica ...

Keywords: Bulk-Synchronous Parallelism, PRAM, reactive programming

14 Controlling interference in ambients

Francesca Levi, Davide Sangiorgi

January 2000 **Proceedings of the 27th ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '00**

Publisher: ACM Press

Full text available: pdf(1.47 MB) Additional Information: full citation, abstract, references, citings, index terms

Two forms of interferences are individuated in Cardelli and Gordon's Mobile Ambients (MA): plain interferences, which are similar to the interferences one finds in CCS and &phgr;-calculus; and grave interferences, which are more dangerous and may be regarded as programming errors. To control interferences, the MA movement primitives are modified. On the new calculus, the Mobile Safe Ambients (SA), a type system is defined t ...

15 Three logics for branching bisimulation

Rocco De Nicola, Frits Vaandrager

March 1995 **Journal of the ACM (JACM)**, Volume 42 Issue 2
Publisher: ACM Press

Full text available: pdf(2.06 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Three temporal logics are introduced that induce on labeled transition systems the same identifications as branching bisimulation, a behavioral equivalence that aims at ignoring invisible transitions while preserving the branching structure of systems. The first logic is an extension of Hennessy-Milner Logic with an "until" operator. The second one is another extension of Hennessy-Milner Logic, which exploits the power of backward modalities. The third logic is CTL* without the ...

Keywords: CTL*, Hennessy-Milner logic, Kripke structures, backward modalities, branching bisimulation equivalence, concurrency, doubly labeled transition systems, labeled transition systems, reactive systems, semantics, stuttering equivalence, until operations

16 A logical analysis of aliasing in imperative higher-order functions
Martin Berger, Kohei Honda, Nobuko Yoshida
September 2005 **ACM SIGPLAN Notices , Proceedings of the tenth ACM SIGPLAN international conference on Functional programming ICFP '05**, Volume 40 Issue 9
Publisher: ACM Press

Full text available: pdf(234.59 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a compositional program logic for call-by-value imperative higher-order functions with general forms of aliasing, which can arise from the use of reference names as function parameters, return values, content of references and parts of data structures. The program logic extends our earlier logic for alias-free imperative higher-order functions with new modal operators which serve as building blocks for clean structural reasoning about programs and data structures in the presence of al ...

Keywords: π-calculus, aliasing, functional programming, hoare-logics, modalities, pointers, typing

17 From process logic to program logic
Kohei Honda
September 2004 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN international conference on Functional programming ICFP '04**, Volume 39 Issue 9
Publisher: ACM Press

Full text available: pdf(208.11 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a process logic for the p -calculus with the linear/affine type discipline [6, 7, 31, 32, 33, 59, 60]. Built on the preceding studies on logics for programs and processes, simple systems of assertions are developed, capturing the classes of behaviours ranging from purely functional interactions to those with destructive update, local state and genericity. A central feature of the logic is representation of the behaviour of an environment as the dual of that of a process in an assertio ...

Keywords: π-calculus, duality, higher-order functions, hoare logic, mobile processes, types

18 An interview with Robin Milner

Karen A. Frenkel

January 1993 **Communications of the ACM**, Volume 36 Issue 1

Publisher: ACM Press

Full text available: pdf(745.11 KB) Additional Information: full citation, index terms

19 Poster session: Deriving refactorings for aspectJ

Leonardo Cole, Paulo Borba

October 2004 **Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications OOPSLA '04**

Publisher: ACM Press

Full text available: pdf(64.67 KB) Additional Information: full citation, abstract, references, index terms

In this paper we present aspect-oriented programming laws that are useful for deriving refactorings for AspectJ. The laws help developers to verify if the transformations they define preserve behavior. We illustrate that by deriving several AspectJ refactorings. We also show that our laws are useful for restructuring two Java applications with the aim of using aspects to modularize common crosscutting concerns.

Keywords: aspect-oriented programming, aspectJ, refactoring, separation of concerns

20 Termination, deadlock, and divergence

L. Aceto, M. Hennessy

January 1992 **Journal of the ACM (JACM)**, Volume 39 Issue 1

Publisher: ACM Press

Full text available: pdf(2.56 MB) Additional Information: full citation, abstract, references, citings, index terms, review

In this paper, a process algebra that incorporates explicit representations of successful termination, deadlock, and divergence is introduced and its semantic theory is analyzed. Both an operational and a denotational semantics for the language is given and it is shown that they agree. The operational theory is based upon a suitable adaptation of the notion of bisimulation preorder. The denotational semantics for the language is given in terms of the initial continuous algebra that satisfie ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next